Claims:

1. An apparatus for protecting a computing device from attacks during operation of the computing device, the apparatus comprising:

an input/output unit,

a control unit coupled to the input/output unit,

an execute unit coupled to the control unit,

a first memory area including memory that is accessible by a user of the computing device, and

a second memory area including memory that is not accessible by the user, the second memory area being configured to store a plurality of return addresses and stack pointers.

2. The apparatus of claim 1, wherein the execute unit is operable to execute a plurality of operations including:

a first operation which stores a first return address in the first memory area and in the second memory area,

a second operation which compares the first return address with a second return address retrieved from the first memory area, and

a third operation which generates an exception if the comparison indicates a mismatch between the first return address and the second return address.

3. The apparatus of claim 1, further comprising a third memory area including memory that is not accessible by a computer user, the third memory area being configured to store a plurality of return addresses and stack pointers.

4. The apparatus of claim 3, wherein the execute unit is operable to execute a plurality of operations including:

a first operation that stores a first return address in the first memory area and in the second memory area,

a second operation that copies the first return address to the third memory area if the second memory area is full,

a third operation that retrieves the first return address from the third memory area,

a fourth operation that compares the first return address with a second return address retrieved from the first memory area, and

a fifth operation that generates an exception if the comparison indicates a

mismatch between the first return address and the second return address.

5. A computing device comprising the apparatus of claim 1.

6. A computing device, comprising:

means for receiving data and programming instructions,

means for processing the data according to the instructions,

means for storing return addresses generated by the means for processing in a first memory area,

means for storing the return addresses in a second memory area not accessible by computer users, and

means for evaluating a return address from the first memory area and a return address from the second memory area to determine whether an attack on a return address has occurred.

7. The computing device of claim 6, further comprising:

means for generating an exception if the means for evaluating determines that an attack has occurred.

8. A computer-readable medium comprising instructions that operate to prevent attacks on return addresses during execution of a computer program, the instructions being executable to:

store a first return address in a first memory area,

store the first return address in a second memory area that is not accessible by computer users,

retrieve a second return address from the first memory area,

compare the first return address and the second return address, and

generate an exception if the first return address is different from the second return address.

9. A computer-readable medium, for use in connection with a computing device, the computer-readable medium including a plurality of instructions that when executed protect the computing device from attacks on return addresses, at least a portion of the computer-readable medium comprising a first memory which is:

configured to store a plurality of return addresses during execution of a computer program,

protected from access by users of the computing device during execution of the computer program, and

accessed by instructions that compare the plurality of return addresses with return addresses stored in a second memory in the computing device.

10. A method of preventing attacks on return addresses during execution of a computer program on a computing device, the method comprising the steps of:

storing a first return address in a first memory that is accessible to computer users and in a second memory that is not accessible to computer users,

retrieving a second return address from the first memory,

comparing the first return address and the second return address, and

generating an exception if the results of the comparing step indicate that an attack has been attempted.

11. The method of claim 10, wherein the step of generating an exception includes generating a hardware exception.

12. The method of claim 10, wherein the storing step is performed if a call instruction is encountered in the computer program.

13. The method of claim 12, wherein the retrieving, comparing, and generating, steps are performed if a return instruction is encountered in the computer program.

14. The method of claim 10, wherein the comparing step is performed at the time of a return instruction commit.

15. The method of claim 10, wherein the comparing step includes the steps of:

recognizing when a data port is not available to accomplish the comparison, and

stalling issuing instructions until a data port is available.

16. The method of claim 10, wherein the storing step includes the step of copying the first return address from the second memory into a third memory that is not accessible by computer users.

17. The method of claim 10, further comprising the step of copying at least a portion of the contents of the second memory into a third memory that is not accessible to computer users if a context switch instruction is encountered in the computer program.

18. The method of claim 17, further comprising the step of copying at least a portion of the contents of the third memory into the second memory.

19. The method of claim 10, further comprising the step of comparing at least a portion of the contents of the first memory with at least a portion of the contents of the second memory if a jump instruction is encountered in the computer program.

20. The method of claim 10, further comprising the step of inserting a random number into the first memory if a jump instruction is encountered in the computer program.